INF 1010
Estruturas de Dados Avançadas

Tabelas de dispersão

outra implementação de Mapa...

Tabelas de dispersão

motivação:

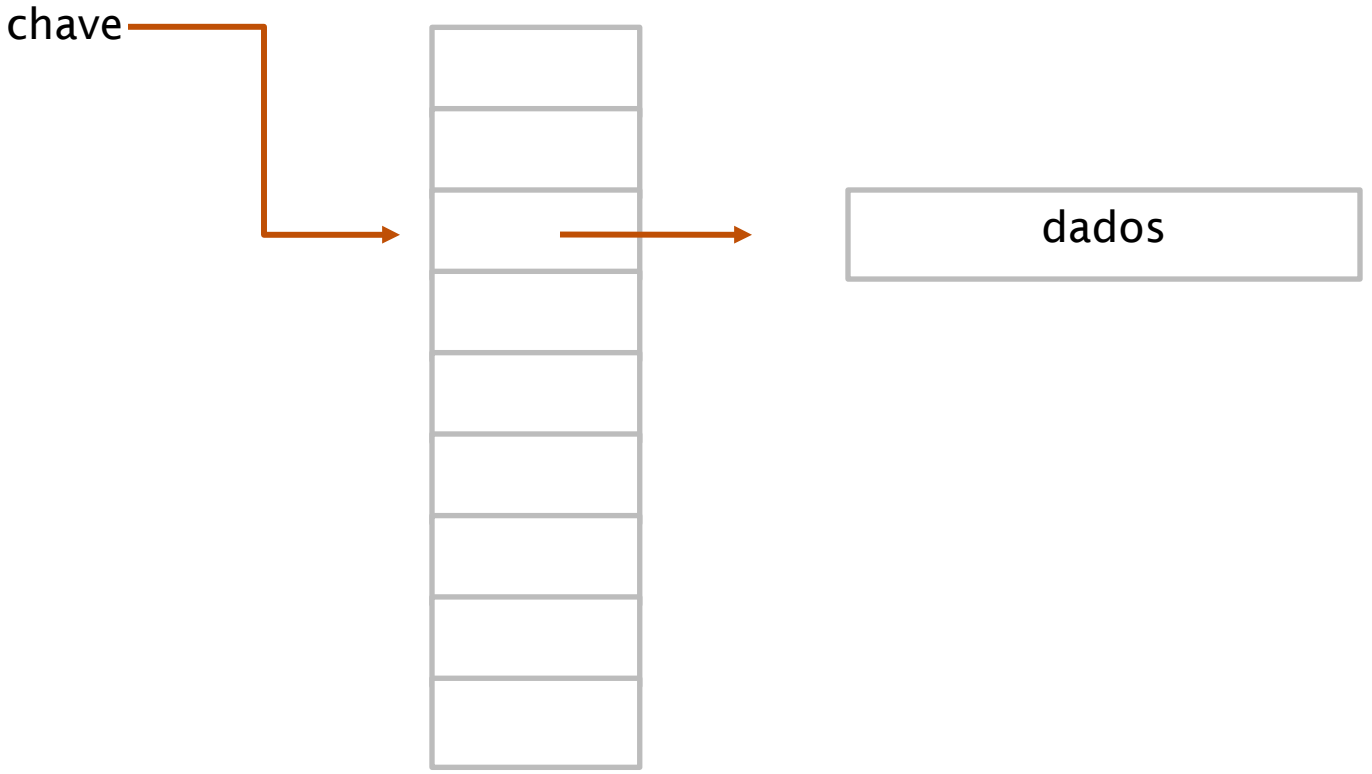
acesso em O(1)

estratégia:

uso de array

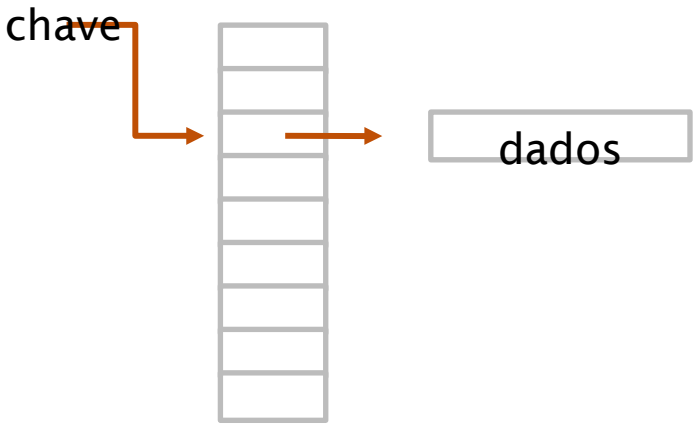
tradeoff: memória X tempo de acesso

acesso direto - arrays



acesso direto

- mas e se
  - as chaves não formam um conjunto contínuo de inteiros?
  - se temos muitas chaves
  - se as chaves não são nem inteiras



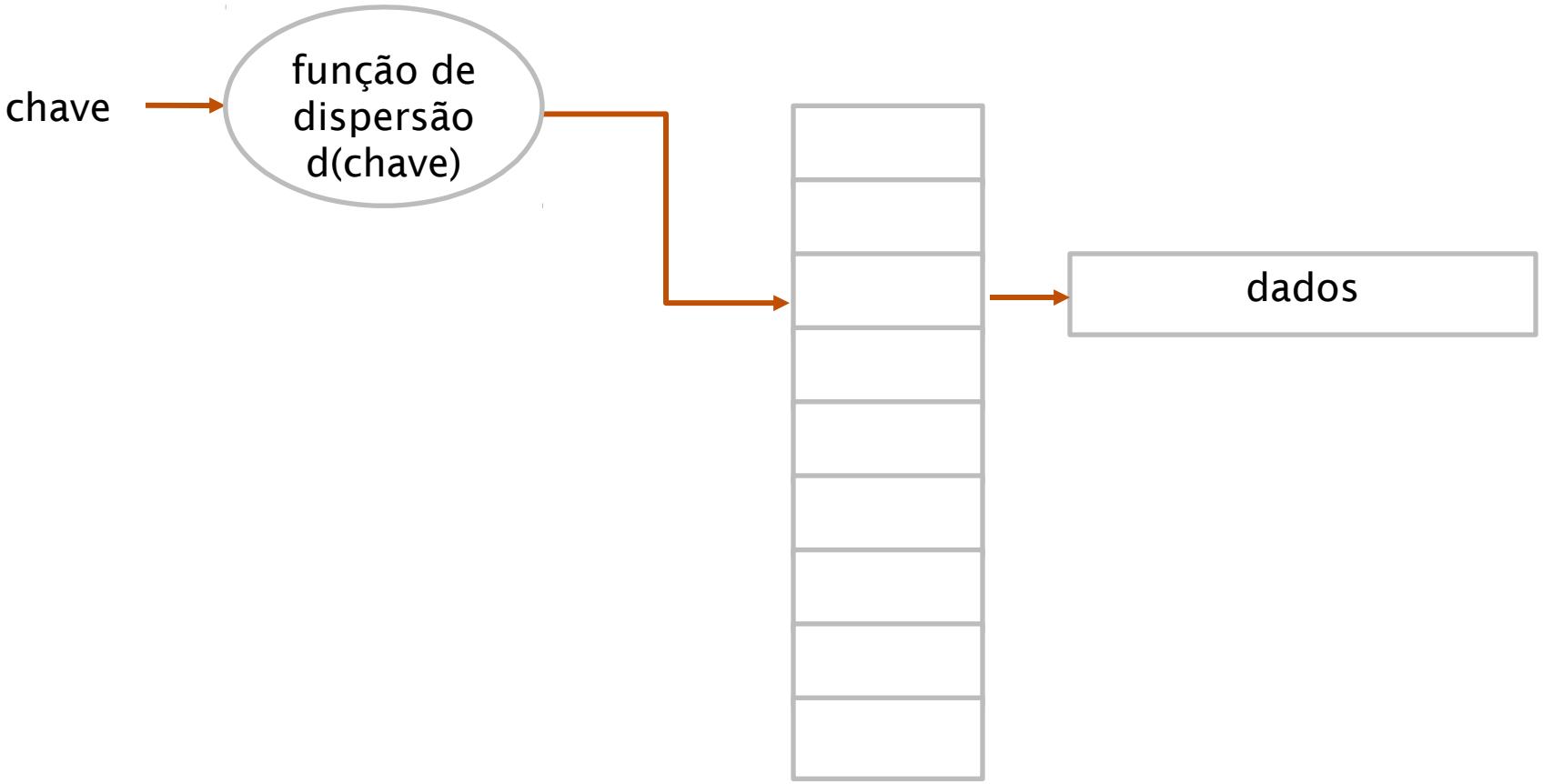
tabelas de dispersão

mapeamento de chaves a posições de array

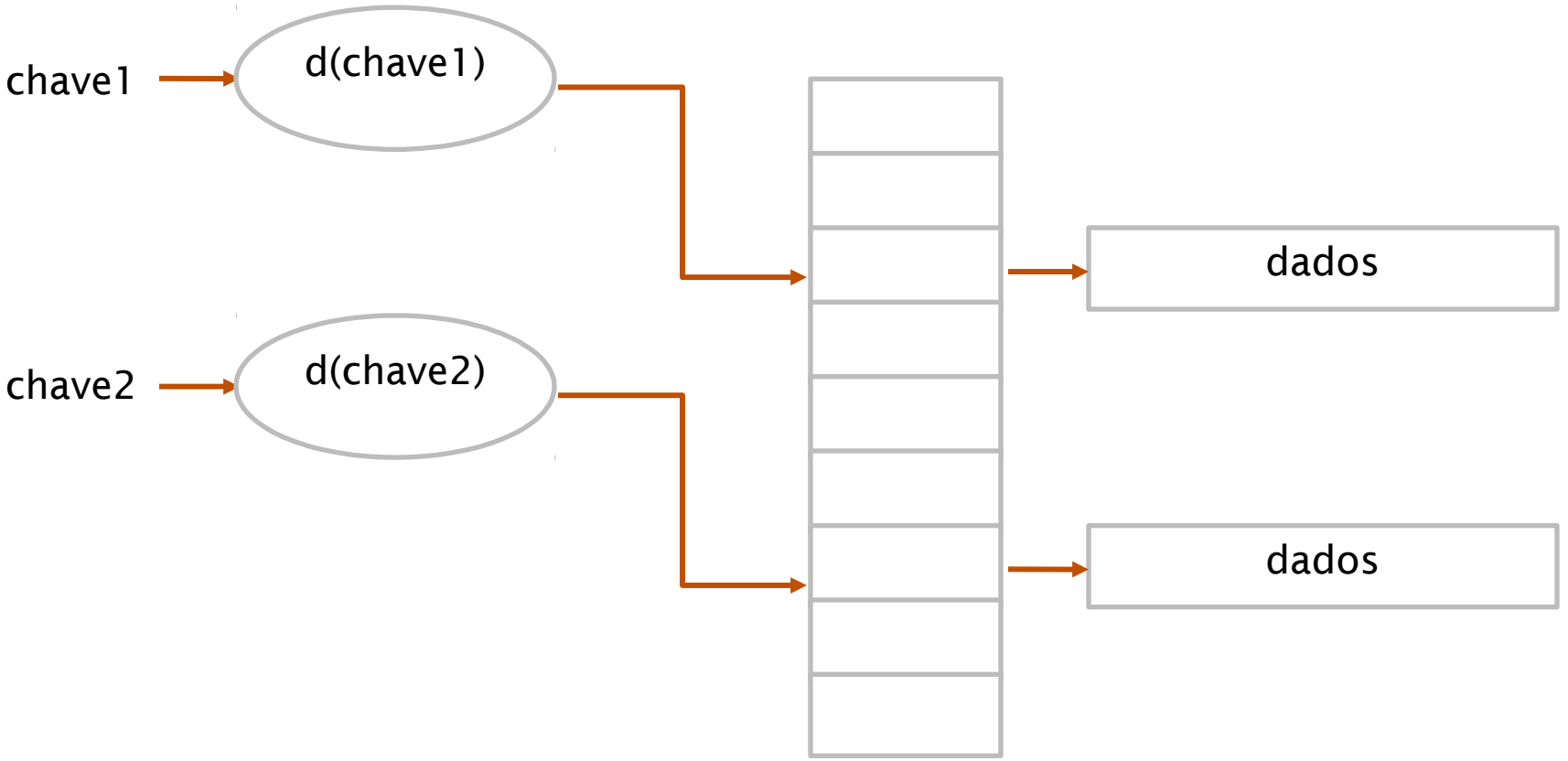
tabelas de dispersão

mapeamento de chaves a posições de array

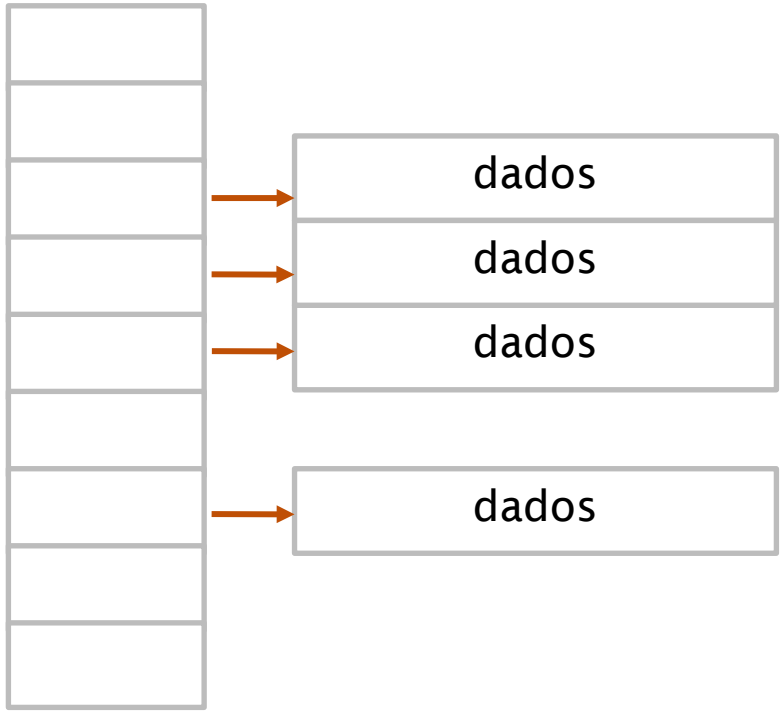
funções de dispersão



boas funções de dispersão



posições livres e ocupadas



características de uma boa função de dispersão

- cálculo “barato”
- poucas colisões

funções de dispersão comuns

- resto da divisão – módulo N (tam. tabela)
  - na prática: N sem fatores primos menores que 20
- mid-square
  - $q = chave^2$
  - toma-se x bits do “meio” de q
- folding
  - chave particionada em blocos de bits
  - blocos “somados”

transformação de chaves não numéricas

- várias formas podem ser usadas:
  - soma de valores ascii
  - soma com shift
  - ...

funções de transformação - exemplo

```
#define Multiplier -1664117991L

int Hash (char* s, int size) {
  int i; unsigned long hashCode;

  hashCode = 0;
  for (i=0; s[i]!=0; i++) {
    hashCode = hashCode*Multiplier + s[i];
  }
  return (hashCode%size);
}
```

colisões

- qualidade da função de dispersão
- tamanho de tabela de dispersão

tratamento de colisões

- uso de outras posições do array
- listas encadeadas

tratamento de colisões

- uso de outras posições do array
-
  - procura linear
  - procura com função de incremento
- listas encadeadas

tratamento de colisões – busca de posições livres

- procura sequencialmente
-
  - *clustering*

```
poscandidata = hash(chave);
while (ocupada(poscandidata)) {
  poscandidata = (poscandidata + 1)%tamtabela;
}
```

tratamento de colisões – busca de posições livres

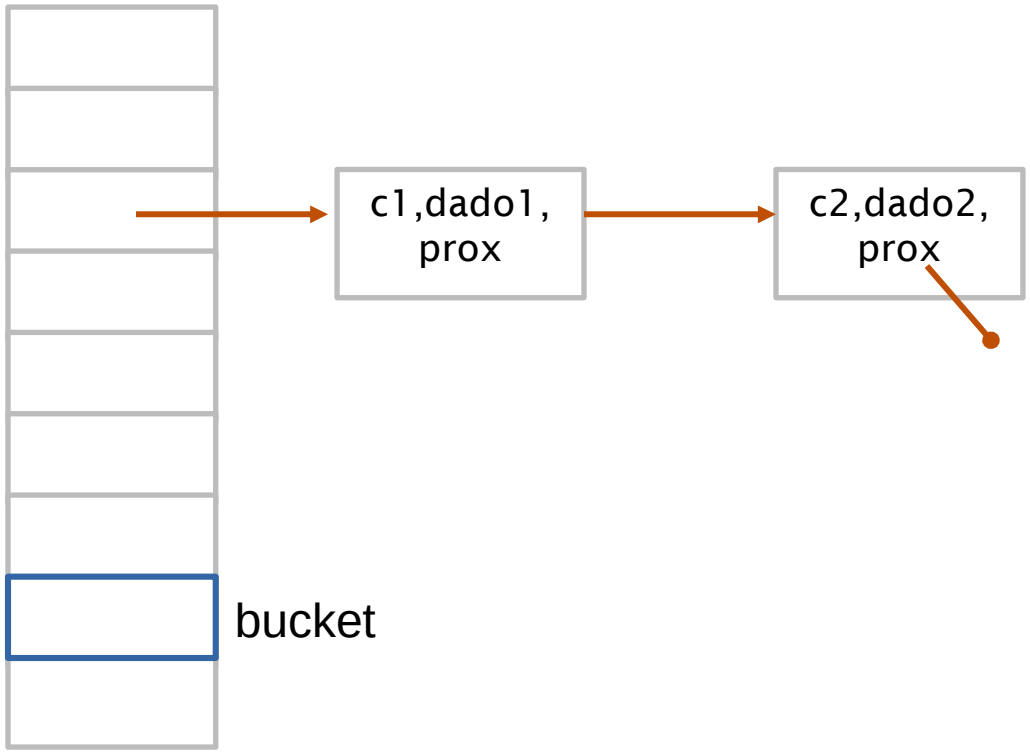
- procura com função de incremento
-
  - evita *clustering*
  -

```
poscandidata = hash(chave);
passo = hash2(chave);
while (ocupada(poscandidata)) {
  poscandidata = (poscandidata + passo)%tamtabela;
}
```

tratamento de colisões

- uso de outras posições do array
- listas encadeadas

tratamento de colisões



- listas encadeadas para itens com mesmo valor de hash

Referências

Celes, Cerqueira, Rangel. Introdução a Estruturas de Dados (2004). Cap. 18.

Kruse, Tondo, Leung, Mogalla. Data Structures and Program Design in C (2007). 8.6.